Attorney's Docket No.: 006570.P009

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application for:	
Hans-Christoph Rohlad, et al.	Examiner: Farhad Ali
Serial No. 10/750,002	Art Unit: 2146
Filed: December 30, 2003	
For: **CLUSTER ARCHITECTURE HAVING A STAR TOPOLOGY WITH CENTRALIZED SERVICES**	

Mail Stop Appeal Brief – Patents
Commissioner For Patents
P.O. Box 1450
Alexandria, V.A. 22313-1450

APPEAL BRIEF

Dear Sir:

Applicant ("Appellant") submits the following Appeal Brief pursuant to 37 C.F.R. §41.37(c) for consideration by the Board of Patent Appeals and Interferences. A payment in the amount of $510.00 was submitted with the Notice of Appeal filed on July 14, 2008, (received by the Patent Office on July 17, 2008) as required by 37 C.F.R. §41.20(b)(1). A payment in the amount of $510.00 is submitted herewith as required by 37 C.F.R. §41.20(b)(2).

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

Hans-Christoph Rohland and Frank Kilian are named as the inventors on the application. Hans-Christoph Rohland and Frank Kilian transferred their rights in the subject application through an assignment executed on March 23, 2004, to SAP Aktiengesellschaft ("SAP AG"), a Corporation of Germany, having a principal place of business at Walldorf, Germany. The assignment is recorded at reel/frame number 015517/0382. Accordingly, SAP AG is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences that will directly affect, be directly affected by or have a bearing on the Board's decision in this Appeal.

III. STATUS OF CLAIMS

Claims 1-25 are pending in the application. The Examiner has rejected claims 1-25. Appellant respectfully appeals the rejection of claims 1-25.

IV.STATUS OF AMENDMENTS

Appellant has submitted an amendment of claims 6 and 18 in a Response to Final Office Action mailed on June 9, 2008. Examiner failed to indicate in the Advisory Action whether the submitted claim amendments had been entered. Appellant assumes that these amended claims have been entered as they do not raise new issues that would require further search.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Embodiments of the invention relate to a star topology cluster architecture for application servers. See Specification, Page 1, Lines 5-7.

Independent claim 1 recites a system including the elements of a database (see Specification, Page 3, Lines 6-7 and Figure 1, 114); a message server with no persistent state (see Specification, Page 3, Lines 27-28 and Figure 1, 110); and a plurality of instances of an application server (see Specification, Page 2, Lines 17-19 and Figure 1, 104). The instances of an application server implement a Java application model coupled in a star topology with the message server at a center of the star topology. See Specification, Page 2, Lines 17-19. The instances of an application server share the database. See Specification, Page 3, Lines 6-7 and Figure 1, 114.

Claim 2 depends from claim 1 and recites the further limitations of a dispatcher node and a plurality of server nodes. See Specification, Page 3, Lines 2-3.

Claim 3 depends from claim 2 and recites the further limitations of a java 2 enterprise edition (J2EE) engine. See Specification, Page 2, Lines 30-33.

Claim 4 depends from claim 1 and recites the further limitations of a central lock server to provide cluster wide locks to the plurality of instances. See Specification, Page 3, Lines 31-34 and Figure 1, 112.

Claim 5 depends from claim 1 and recites the further limitations of a first data structure to store a list of connected clients and a second data structure and a list of services provided in the system. See Specification, Page 3, Lines 6-13.

Independent claim 6 recites a computer readable storage media containing executable computer program instructions which when executed cause a digital processing system to perform a method. The method includes the elements of starting a central services node to provide at least one of a locking service and a messaging service (see Specification, Page 5, Lines 6-7 and Figure 3, 302); starting a plurality of application server instances (see Specification, Page 5, Lines 9-10 and Figure 3, 304); and organizing the application server instances into a cluster having star topology (see Specification, Page 2, Lines 17-19 and Figure 1, 104). In the star topology of server instances, the central services node is at the center. See Id. Further, the messaging service has no persistent state. See Specification, Page 3, Lines 27-28 and Figure 1, 110.

Claim 7 depends from claim 6 and recites the further limitations of sharing a database among the plurality of application server instances. See Specification, Page 3, Lines 6-7 and Figure 1, 114.

Claim 8 depends from claim 6 and recites the further limitations of starting, for each application server instance of the plurality, a dispatcher node and a plurality of server nodes. See Specification, Page 3, Lines 2-3.

Claim 9 depends from claim 6 and recites the further limitations of starting a message server having no persistent state. See Specification, Page 3, Lines 27-28 and Figure 1, 110.

Claim 10 depends from claim 6 and recites the further limitations of registering each application server with the messaging server. See Specification, Page 5, Lines 12-13 and Figure 3, 306.

Claim 11 depends from claim 6 and recites the further limitations of conducting inter instance communication through the messaging service. See Specification, Page 3, Lines 14-17.

Claim 12 depends from claim 9 and recites the further limitations of restarting the message server without state recovery responsive to a system failure. See Specification, Page 5, Lines 22-27 and Figure 3, 312.

Claim 13 depends from claim 10 and recites the further limitations of notifying all registered instances from the message server when an additional instance joins the cluster. See Specification, Page 5, Lines 16-21 and Figure 3, 308.

Independent claim 14 recites a system including a means for organizing a plurality of application servers instances into a cluster having a star topology with a central services node at a center of the star topology (see Specification, Page 2, Lines 17-20 and Figure 1, 102); a means for sharing a storage resource across the cluster (see Specification, Page 3, Lines 6-7 and Figure 1, 114); and a means for performing centralized inter instances communication without maintenance of persistent state information (see Specification, Page 3, Lines 14-17).

Claim 15 depends from claim 14 and recites the further limitations of means for centralized locking of a resource within the cluster. See Specification, Page 3, Lines 25-27 and Figure 1, 112.

Claim 16 depends from claim 14 and recites the further limitations of a message server having no persistent state. See Specification, Page 3, Lines 27-28 and Figure 1, 110.

Claim 17 depends from claim 14 and recites the further limitations of means for registering instances and means for recording services provided in the cluster. See Specification, Page 5, Lines 12-13 and Figure 3, 306.

Independent claim 18 recites a method including starting a central services node to provide at least one of a locking service and a messaging service (see Specification, Page 3, Lines 6-7 and Figure 1, 114); starting a plurality of application server instances (see Specification, Page 5, Lines 9-10 and Figure 3, 304); and organizing the application server instances into a cluster having star topology with the central services node at a center of the star topology (see Specification, Page 2, Lines 17-19 and Figure 1, 104). Further, the messaging service does not maintain a persistent state. See Specification, Page 3, Lines 27-28 and Figure 1, 110.

Claim 19 depends from claim 18 and recites the further limitations of sharing a database among the plurality of application server instances. See Specification, Page 3, Lines 6-7 and Figure 1, 114.

Claim 20 depends from claim 18 and recites the further limitations of starting, for each instance of the plurality, a dispatcher node and a plurality of server nodes. See Specification, Page 3, Lines 2-3.

Claim 21 depends from claim 18 and recites the further limitations of starting a message server having no persistent state. See Specification, Page 3, Lines 27-28 and Figure 1, 110.

Claim 22 depends from claim 18 and recites the further limitations of registering each application server with the messaging server. See Specification, Page 5, Lines 12-13 and Figure 3, 306.

Claim 23 depends from claim 18 and recites the further limitations of conducting inter instance communication through the messaging service. See Specification, Page 3, Lines 14-17.

Claim 24 depends from claim 21 and recites the further limitations of restarting the message server without state recovery responsive to a system failure. See Specification, Page 5, Lines 22-27 and Figure 3, 312.

Claim 25 depends from claim 22 and recites the further limitations of notifying all registered instances from the message server when an additional instance joins the cluster. See Specification, Page 5, Lines 16-21 and Figure 3, 308.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

In the Final Office Action dated April 14, 2008, the Examiner has rejected claims 1-25.

Claims 6-8, 10-11, 13, 18-20, 22-23 ad 25 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,954,757 issued to Zargham et al. (hereinafter "Zargham").

Claims 1-5, 9, 12, 14-17, 21, and 24 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Zargham in view of U.S. Patent Publication No. 2002/0078132 filed by Cullen et al. (hereinafter "Cullen").

All of the claims do not stand or fall together. The basis for the separate patentability of the claims is set forth below.

VII. ARGUMENT

A. Overview of the Prior Art – Zargham

Zargham proposes a framework, architecture and system that reduce the latency of business transactions. See Zargham, Abstract. Zargham seeks to allow the real-time integration and availability of services, applications and data throughout an enterprise system. See Zargham, Column 9, Lines 34-43. To facilitate the real time availability of data and operations, Zargham utilizes a central repository, database extractors, database loaders, and application adapters. See Zargham, Column 2, Lines 42-47.

However, Zargham is silent with regard to the use and storage of message server state information. Further, Zargham fails to disclose registering application servers as they join the cluster and notifying other nodes that an instance has joined the cluster.

B. Overview of the Prior Art – Cullen

Cullen proposes a method of handling messages which are received at a message server. See Cullen, Paragraph [0005]. Further, Cullen improves reliability of message delivery by utilizing persistent storage in the message handling method. See Id. The method includes storing a message in non-persistent storage until a specified time interval has been reached. See Cullen, Paragraph [0017]. Thereupon, the message is moved into persistent storage. See Id. Saving messages in persistent storage improves message delivery reliability by eliminating the possibility of message loss upon a storage failure. See Cullen, Paragraph [0025].

Although messages are stored in non-persistent storage for a period of time before being transferred to persistent storage, Cullen does not reveal the method or location of message server state data. Further, Cullen fails to disclose registering application servers as they join the cluster and notifying other nodes that an instance has joined the cluster.

C. Rejection of Claims 6-8, 10-11, 13, 18-20, 22-23 and 25 Under 35 U.S.C. §102

Claims 6-8, 10-11, 13, 18-20, 22-23 ad 25 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,954,757 issued to Zargham et al. (hereinafter "Zargham").

To anticipate a claim, a single reference must disclose each element of that claim. Thus, "[t]he identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co.,* 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). Also, "[t]he elements must be arranged as required by the claim." *See In re Bond,* 910 F.2d 831, 15 USPQ2d 1566 (Fed. Cir. 1990) and MPEP § 2131.

However, as discussed below, the cited reference fails to disclose each element of claims 6-8, 10-11, 13, 18-20, 22-23 and 25.

1. Claims 6, 7, 11, 18, 19, and 23

a) Independent Claims 6 and 18 Are Patentable at Least Because Zargham Fails to Disclose a Messaging Service Having No Persistent State.

Independent claims 6 and 18 were amended by Appellant in the Response to the Final Office Action mailed on June 9, 2008. However, these amendments were not acknowledged as entered by the Examiner in the Advisory Action mailed June 30, 2008.

Amended claims 6 and 18 include the elements of "the messaging service having no persistent state" (claim 6) and "the messaging service not maintaining a persistent state" (claim 18). The Examiner has acknowledged that Zargham "does not disclose a message server having no persistent state." See Final Office Action mailed April 14, 2008, Page 8. Therefore, Zargham fails to disclose each element of claims 6 and 8, and therefore cannot form the basis of a rejection under 35 U.S.C. § 102.

Thus, in view of at least the foregoing reasons, claims 6 and 18 are directed toward allowable subject matter and are separately patentable. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 6 and 18 be overturned.

Further, claims 6 and 18 include elements analogous to those of claims 1 and 14, such as "the messaging service having no persistent state" (claim 6) and "the messaging service not maintaining a persistent state" (claim 18). For at least the reasons discussed below in Appellant's argument over the 35 U.S.C. § 103 rejection of claims 1 and 14, Appellant submits that Cullen does not cure the deficiencies of Zargham. Consequently, Zargham in view of Cullen fails to teach or suggest each element in claims 6 and 18.

b) Dependent Claims 7, 11, 19, and 23 Depend from Respective Patentable Base Claims.

Dependent claims 7, 11, 19, and 23 depend from base claims 6 and 18, respectively and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with the respective base claims, Zargham fails to teach each element of claims 7, 11, 19, and 23. Therefore, claims 7, 11, 19, and 23 are patentable over the art of record because each of these claims depends from either claim 6 or 18.

Thus, in view of at least the foregoing reasons, claims 7, 11, 19, and 23 are directed toward allowable subject matter and are separately patentable. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 7, 11, 19, and 23 be overturned.

2. Claims 10 and 22

a) Dependent Claims 10 and 22 Depend from Respective Patentable Base Claims.

Claims 10 and 22 depend from claims 6 and 18, respectively, and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with claims 6 and 18, Zargham fails to teach each element of claims 10 and 22. Therefore, claims 10 and 22 are patentable over the art of record because each of these claims depends from claim 6 or 18.

Thus, in view of at least the foregoing reasons, claims 10 and 22 are directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 10 and 22 be overturned.

b) Dependent Claims 10 and 22 Are Patentable at Least Because Zargham Fails to Disclose Registering Each Application Server With the Messaging Server.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claims 10 and 22.

Claims 10 and 22 recite the elements of "registering each application server with the messaging server." In the Final Office Action, the Examiner cited column 13, lines 66-68 of Zargham as allegedly teaching these elements. See Final Office Action, pages 4 and 7. However, Zargham fails to teach these elements. The sections of Zargham relied on by the Examiner teach the availability of publishing and subscribing operations in a zero latency environment. See Zargham, Column 13, Lines 66-68. However, these sections of Zargham do not teach using this message distribution system for any particular purpose. See Id. In contrast, claims 10 and 22 disclose the transfer of messages for registering application servers with a messaging server. Disclosing that a method for delivering messages is available does not disclose a particular transfer of messages between a distinct set of objects. Further, the Examiner has failed to provide an explanation regarding how the availability of publish and subscribe teach "registering each application server with the messaging server." Thus, the cited sections of Zargham have not disclosed "registering each application server with the messaging server."

Further, Appellant has been unable to discern any portion of Zargham that teach these elements. Therefore, Zargham fails to disclose each element of claims 10 and 22, and therefore cannot form the basis of a rejection under 35 U.S.C. § 102.

Thus, in view of at least the foregoing additional reasons, claims 10 and 22 are separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 10 and 22 be overturned.

3. Claims 13 and 25

a) Dependent Claims 13 and 25 Depend from Respective Patentable Base Claims.

Claims 13 and 25 depend from claims 6 and 18, respectively, and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with claims 6 and 18, Zargham fails to teach each element of claims 13 and 25. Therefore, claims 13 and 25 are patentable over the art of record because each of these claims depends from claim 6 or 18.

Thus, in view of at least the foregoing reasons, claims 13 and 25 are directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 13 and 25 be overturned.

b) Dependent Claims 13 and 25 Are Patentable at Least Because Zargham Fails to Disclose Notifying Registered Instances When an Additional Instance Joins the Cluster.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claims 13 and 25.

Claims 13 and 25 recite the elements of "notifying all registered instances from the message server when an additional instance joins the cluster." In the Final Office Action, the Examiner has cited column 13, lines 66-68 of Zargham as allegedly teaching these elements. See Final Office Action, pages 5 and 7. However, Zargham fails to teach these elements. The sections of Zargham relied on by the Examiner teach publishing and subscribing operations in a zero latency environment. See Zargham, Column 13, Lines 66-68. However, these sections of Zargham do not teach using this message distribution system for any particular purpose. See Id. In contrast, claims 13 and 25 disclose the notification of registered instances from the message servers when a new instance joins the cluster. Disclosing that a method for delivering messages is available does not disclose a particular transfer of messages between a distinct set of objects, specifically from a name server to registered instances. Further, the Examiner has failed to provide an explanation regarding Zargham teaches the timing of sending notification "when an additional instance joins the cluster." Therefore, the Examiner has failed to establish that Zargham discloses each element of claims 13 and 25, and therefore cannot form the basis of a rejection under 35 U.S.C. § 102.

Thus, in view of at least the foregoing additional reasons, claims 13 and 25 are separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 13 and 25 be overturned.

4. Claims 8 and 20

a) Dependent Claims 8 and 20 Depend from Respective Patentable Base Claims.

Claims 8 and 20 depend from claims 6 and 18, respectively, and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with claims 6 and 18, Zargham fails to teach each element of claims 8 and 20. Therefore, claims 8 and 20 are patentable over the art of record because each of these claims depends from claim 6 or 18.

Thus, in view of at least the foregoing reasons, claims 8 and 20 are directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 8 and 20 be overturned.

b) Dependent Claims 8 and 20 Are Patentable Over the Cited Art Because Zargham Fails to Teach a Dispatcher Node.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claims 8 and 20.

Claims 8 and 20 include a "dispatcher node." Zargham fails to teach these elements. The Examiner cites Figure 9 in Zargham to teach this element of claims 8 and 20. However, none of the components depicted in Figure 9 or in the corresponding description in the Specification appear to disclose a dispatcher node as argued by the Examiner. The term "dispatcher node" does not appear in Figure 9 or in the relative sections of Zargham. Further, the Examiner has not established how Zargham teaches these elements of claims 8 and 20. The reference cited by the Examiner discloses a zero latency enterprise framework. See Zargham, Figure 9. However, none of the components explicitly disclose a dispatcher node. Further, the Examiner has not provided a line of reasoning which explains why Figure 9 would implicitly disclose a dispatcher node.

Moreover, Appellant has been unable to find any sections of Zargham that disclose these elements. Consequently, in view of at least these reasons, Zargham fails to teach each element in claims 8 and 20.

Thus, in view of at least the foregoing additional reasons, claims 8 and 20 are separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 102 rejection of claims 8 and 20 be overturned.

Further, claims 8 and 20 include elements analogous to those of claim 2, such as a dispatcher node. For at least the reasons discussed below in Appellant's argument over the 35 U.S.C. § 103 rejection of claim 2, Appellant submits that Cullen does not cure the deficiencies of Zargham. Consequently, Zargham in view of Cullen fails to teach or suggest each element in claims 8 and 20.

D. Rejection of Claims 1-5, 9, 12, 14-17, 21, and 24 Under 35 U.S.C. § 103

Claims 1-5, 9, 12, 14-17, 21, and 24 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Zargham in view of U.S. Patent Publication No. 2002/0078132 filed by Cullen et al. (hereinafter "Cullen").

To establish a *prima facie* case of obviousness the Examiner must set forth a clear articulation of the reasons that the claimed invention would have been obvious. The reasoning cannot be based on mere conclusory statements. See KSR Int'l Co. v. Teleflex Inc. (KSR), 82 USPQ2d 1385, 1396 (2007) and MPEP § 2142. Further, the Federal Circuit has clarified that the determination of the proper combination of prior art teachings in light of the Supreme Court's decision in KSR Int'l Co. v. Teleflex Inc. is to be based on the flexible application of the teaching, suggestion and motivation (TSM) test, because "as the Supreme Court suggests, a flexible approach to the TSM test prevents hindsight and focuses on evidence before the time of the invention." In re Translogic Tech., Inc., 504 F.3d 1249, 1257 (Fed. Cir. 2007).

However, as discussed below, the cited references fail to teach or suggest each element of claims 1-5, 9, 12, 14-17, 21, and 24.

1. **Claims 1, 4 and 14-17**

a) Independent Claims 1 and 14 Are Patentable at Least Because <u>Zargham</u> in View of <u>Cullen</u> Fails to Teach or Suggest a Message Server Having No Persistent State.

Claims 1 and 14 recite the elements of "a message server having no persistent state data" (claim 1) and "means for performing centralized inter instances communication without maintenance of persistent state information" (claim 14). The Examiner has acknowledged that <u>Zargham</u> "does not disclose a message server having no persistent state." <u>See</u> <u>Final Office Action</u> mailed April 14, 2008, Page 8. Instead the Examiner relies on <u>Cullen</u> to cure the deficiencies of <u>Zargham</u>. <u>See</u> <u>Final Office Action</u> mailed April 14, 2008, Page 9. <u>Cullen</u> discloses a method of initially storing messages in non-persistent storage and subsequent to a delay interval moving the message to persistent storage. <u>See</u> <u>Cullen,</u> Paragraph [0005]. The Examiner relies on this portion of <u>Cullen</u> to teach the elements of claims 1 and 14 relating to message server state data. However, <u>Cullen</u> never addresses message server state data. Nevertheless, the Examiner argues that the failure of a reference to disclose the existence of server state data implies that the message server has no persistent state. <u>See</u> <u>Final Office Action</u> mailed April 14, 2008, Page 19. The Examiner is confusing not mentioning a particular piece of data as implicitly disclosing that data is stored in non-persistent memory.

The Examiner has not established that these limitations are inherently taught or suggested by the reference. To establish inherency, the descriptive matter must *necessarily* be present in the cited reference. <u>See</u> <u>MPEP</u> § 2112(IV).

> The fact that a certain result or characteristic <u>may</u> occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic. *In re Rijckaert*, 9 F.3d 1531, 1534, 28 USPQ2d 1955, 1957 (Fed. Cir. 1993) (reversed rejection because inherency was based on what

> would result due to optimization of conditions, not what was necessarily present in the prior art); *In re Oelrich*, 666 F.2d 578, 581-82, 212 USPQ 323, 326 (CCPA 1981). "To establish inherency, the extrinsic evidence 'must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill. Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient.' " *In re Robertson*, 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999) (citations omitted) (The claims were drawn to a disposable diaper having three fastening elements. The reference disclosed two fastening elements that could perform the same function as the three fastening elements in the claims. The court construed the claims to require three separate elements and held that the reference did not disclose a separate third fastening element, either expressly or inherently.)

See MPEP § 2112(IV) (emphasis in original). The aspect of a message server not having persistent state data is not *necessarily* present in Cullen. For example, state data in Cullen could be stored in persistent storage. Cullen mentions both a persistent storage and non-persistent storage but makes no reference to where message server state data is stored. See Cullen, Paragraph [0005]. Thus, message server state data could be stored in either persistent or non-persistent storage. There is no implication that it is necessary to store message server state data in non-persistent storage. Moreover, Cullen infers message server state data would be maintained persistently, because Cullen discloses recovering messages after a failure which would most easily be achieved by the storage of message state data persistently to assist in the recovery process and avoid having to determine the state of each message by other means.

Moreover, in the Advisory Action mailed June 30, 2008, the Examiner interprets Cullen as suggesting the use of non-persistent storage for non-guaranteed delivery of messages. See Advisory Action mailed June 30, 2008, Page 2. The Examiner has not established that Cullen discloses the storage of state data in non-persistent storage as disclosed by claims 1 and 14. See Id. "State data" is not analogous to "message data."

State refers to "a snapshot of the measure of various conditions in the system." "Program state." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 1 July 2008. <http://en.wikipedia.org/wiki/Program_state>. However, a message is defined as an object which conveys a broad selection of data between parties. "Message." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 28 July 2008. <http://en.wikipedia.org/wiki/Message#In_computer_science>. Thus, a state is a depiction of a set of characteristics which accurately represent the status and condition of a system for a particular point in time while a message is a container for transporting data. Accordingly, "state data" and "message data" are not analogous. Since the Examiner has only attempted to show that Cullen teaches the storage of message data in non-persistent storage, the Examiner has failed to make a *prima facie* showing that Cullen teaches or suggests a message server without persistent *state data.*

Additionally, the Examiner argues in the Advisory Action mailed June 30, 2008 that Cullen's use of persistent storage to improve reliability does not negate the possibility that a person of ordinary skill in the art would interpret Cullen as teaching or suggesting utilizing non-persistent storage for state data. See Advisory Action mailed June 30, 2008, Page 2. Utilizing this logic, any negative limitation could be said to be taught by a reference by merely labeling contradictory methods and structures of references that do not fit the negative limitation as "extra features." In fact, Cullen clearly teaches persistently storing at least some messages as backup, as the Examiner admits on page 19 of the Final Office Action. The persistent storage of any message related data means that Cullen cannot meet the recited limitations of a messaging source having no persistent state

Consequently, in view of at least these reasons, Zargham in view of Cullen fails to teach or suggest each element in claims 1 and 14. Thus, in view of at least the foregoing reasons, claims 1 and 14 are directed toward allowable subject matter and are separately patentable. Accordingly, Appellant respectfully requests that the § 103 rejection of claims 1 and 14 be overturned.

b) Dependent Claims 4 and 15-17 Depend from Respective Patentable Base Claims.

Dependent claims 4 and 15-17 depend from base claims 1 and 14, respectively and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with the respective base claims, Zargham in view of Cullen fails to teach or suggest each element of claims 4 and 15-17. Therefore, claims 4 and 15-17 are patentable over the art of record because each of these claims depends from either claim 1 or 14.

Thus, in view of at least the foregoing reasons, claims 4 and 15-17 are directed toward allowable subject matter and are separately patentable. Accordingly, Appellant respectfully requests that the § 103 rejection of claims 4 and 15-17 be overturned.

2. Claims 2 and 3

a) Dependent Claim 2 Depends from a Patentable Base Claim.

Claim 2 depends from claim 1 and incorporates the limitations thereof. Thus, for at least the reasons discussed above in connection with claim 1, Zargham in view of Cullen fails to teach or suggest each element of claim 2. Therefore, claim 2 is patentable over the art of record because this claim depends from claim 1.

Thus, in view of at least the foregoing reasons, claim 2 is directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 103 rejection of claim 2 be overturned.

b) Dependent Claim 2 Is Patentable Over the Cited Art Because Zargham in View of Cullen Fails to Teach or Suggest a Dispatcher Node.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claim 2.

Claim 2 includes a "dispatcher node." Zargham and Cullen fail to teach or suggest this element. The Examiner cites Figure 9 in Zargham to teach this element of claim 2. However, none of the components depicted in Figure 9 or in the corresponding description in the Specification appear to disclose a dispatcher node as argued by the Examiner. The Examiner has not established how Zargham or Cullen teach or suggest this element of claim 2. To determine obviousness of a claim: (1) factual findings must be made under the factors set forth in Graham v. John Deere Co., 383 U.S. 1, 148 USPQ 459 (1966); and (2) the analysis supporting the rejection under 35 U.S.C. § 103 should be made explicit and there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. See MPEP §§ 2141(II), 2141(III), and 2142; KSR International Co. v. Teleflex Inc., 82 USPQ2d 1385, 1396; see e.g., MPEP § 2143 (providing a number of rationales which are consistent with the proper "functional approach" to the determination of obviousness as laid down in Graham). The reference cited by the Examiner discloses a zero latency enterprise framework. See Zargham, Figure 9. However, none of the illustrated components or related sections of the specification explicitly disclose a "dispatcher node." Thus, these sections of Zargham fail to explicitly teach or suggest a "dispatcher node." Further, the Examiner has not provided a line of reasoning which explains why a person of ordinary skill in the art would have found a "dispatcher node" as an obvious component of the zero latency enterprise framework.

Moreover, Appellant has been unable to find any sections of Zargham or Cullen that disclose these elements. Therefore, by failing to provide the reasoning for relying on these portions of Zargham, the Examiner has failed to meet the requirement under KSR to articulate a rationale for showing a *prima facie* case of obviousness of these elements.

Thus, in view of at least these reasons, claim 2 is separately patentable over the art of record. Accordingly, Appellant respectfully requests that the § 103 rejection of claim 2 be overturned.

c) Dependent Claim 3 Depends from Respective Patentable Base Claims.

Dependent claim 3 depends from base claim 2 and incorporates the limitations thereof. Thus, for at least the reasons discussed above in connection with base claim 2, Zargham in view of Cullen fails to teach or suggest each element of claim 3. Therefore, claim 3 is patentable over the art of record because this claim depends from claim 2.

Thus, in view of at least the foregoing reasons, claim 3 is directed toward allowable subject matter and is separately patentable. Accordingly, Appellant respectfully requests that the § 103 rejection of claim 3 be overturned.

3. Claim 5

a) Dependent Claim 5 Depends from a Patentable Base Claim.

Claim 5 depends from claim 1 and incorporates the limitations thereof. Thus, for at least the reasons discussed above in connection with claim 1, Zargham in view of Cullen fails to teach or suggest each element of claim 5. Therefore, claim 5 is patentable over the art of record because this claim depends from claim 1.

Thus, in view of at least the foregoing reasons, claim 5 is directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 103 rejection of claim 5 be overturned.

b) Dependent Claim 5 Is at Least Patentable Over Zargham in View of Cullen Because the Cited Art Fails to Teach or Suggest Storing Clients in a First Data Structure and Storing Services in a Second Data Structure.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claim 5.

Claim 5 recites the elements of "a first data structure to store a list of connected clients; and a second data structure and a list of services provided in the system." In the Final Office Action, the Examiner has cited column 1, Lines 43-46 of Zargham as allegedly teaching or suggesting these elements. However, Zargham fails to teach or suggest these elements. The sections of Zargham relied on by the Examiner teach storing data in a data storage cache. See Zargham, Column 1, Lines 43-46. Storing data in a data storage cache does not teach or suggest the use of distinct data structures for storage of clients and services. "[A] cache is a temporary storage area where frequently accessed data can be stored for rapid access." "Cache." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 23 August 2008. <http://en.wikipedia.org/wiki/Cache>. Claim 5 describes storing data using a particular arrangement of data structures. Storing data in a cache does not infer a particular set of structures for storage of the data. Therefore, for at least these reasons, the elements of "a first data structure to store a list of connected clients; and a second data structure and a list of services provided in the system" are not taught or suggested by Zargham.

Further, the Examiner has failed to cite and Appellant is unable to discern any portion of Cullen that allegedly teaches or suggests the missing elements. Thus, in view of at least the foregoing reasons, Zargham in view of Cullen fails to teach or suggest each element of claim 5. Therefore, the Examiner has failed to establish a *prima facie* case of obviousness.

Thus, in view of at least the reasons set forth above, claim 5 is separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 103 rejection of claim 5 be overturned.

4. Claims 12 and 24

a) Dependent Claim 12 and 24 Depends from Respective Patentable Base Claims.

Claims 12 and 24 depend from claims 6 and 9, respectively, and incorporate the limitations thereof. Thus, for at least the reasons discussed above in connection with claims 6 and 9, Zargham in view of Cullen fails to teach or suggest each element of claims 12 and 24. Therefore, claims 12 and 24 are patentable over the art of record because these claims depend from claim 6 or 9.

Thus, in view of at least the foregoing reasons, claims 12 and 24 are directed toward allowable subject matter. Accordingly, Appellant respectfully requests that the § 103 rejection of claims 12 and 24 be overturned.

b) Dependent Claims 12 and 24 Are at Least Patentable Over Zargham in View of Cullen Because the Cited Art Fails to Teach or Suggest Restarting the Message Server Without State Recovery.

Appellant submits the following additional reasons (in addition to the reasons stated above) to show the separate patentability of claims 12 and 24.

Claims 12 and 24 recite the elements of "restarting the message server without state recovery responsive to a system failure." In the Final Office Action, the Examiner has cited column 18, lines 46-51 of Zargham as allegedly teaching or suggesting these elements. See Final Office Action, Pages 11-12 and 16. However, Zargham fails to teach or suggest these elements. The sections of Zargham relied on by the Examiner teach restarting and managing transactions. See Zargham, Column 18, Lines 46-51. However, these sections of Zargham make no reference to restarting a message server without state data. Zargham indicates that restarting a transaction is possible but makes no indication whether state data is used in the process. See Id.

Moreover, the Examiner has failed to establish that Zargham inherently teaches or suggest these elements. To establish inherency, the descriptive matter must *necessarily* be present in the cited reference. See MPEP § 2112(IV). However, the elements of "restarting the message server without state recovery responsive to a system failure" are not *necessarily* present in Zargham. In fact, the claims 8, 14, 18, 20, and 44 indicate that transaction state information is stored by the system in Zargham. See Zargham, claims 14, 18, 20, and 44. Specifically, claim 8 read together with claim 14 suggest that transaction state information is warehoused in an enterprise wide database. See Zargham, Claims 8 and 14. This stored transaction state data could be most logically used during restart operations because state data provides the system with the most accurate representation of the status of the transaction prior to failure. Although Zargham does not explicitly indicate that state data is used while restarting a transaction, it would be possible to access this state information during a restart and this data would most likely be used. Based on the greater possibility that state information which has been previously stored would also be utilized during a transaction restart, Zargham does not implicitly require that state information not be used during a restart. Thus, Zargham teaches away from the assumption of the Examiner. Therefore, for at least these reasons, the elements of "restarting the message server without state recovery responsive to a system failure" are not necessarily present in Zargham.

Thus, in view of at least the reasons set forth above; claims 12 and 24 are separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 103 rejection of claims 12 and 24 be overturned.

5. Claims 9 and 21

a) Dependent Claims 9 and 21 Are at Least Patentable Over <u>Zargham</u> in View of <u>Cullen</u> Because the Cited Art Fails to Teach or Suggest a Messaging Service Having No Persistent State.

Claims 9 and 21 depend from independent claims 6 and 18 respectively, and incorporate the limitations thereof. The Examiner's argument assumes that Zargham discloses all elements of claims 6 and 18 which are incorporated in dependent claims 9 and 21. However, as discussed above in Appellants argument over the 35 U.S.C. § 102 rejections of claims 6 and 18, Zargham does not disclose all these limitations. Furthermore, Examiner has not provided any argument regarding Cullen coverage of these limitations. Thus, the combination of Zargham and Cullen do not teach or suggest each element of claims 9 and 21. Accordingly, Applicants respectfully request reconsideration and withdrawal of the rejection of these claims.

Thus, in view of at least the reasons set forth above; claims 9 and 21 are separately patentable over the cited art. Accordingly, Appellant respectfully requests that the § 103 rejection of claims 12 and 24 be overturned.

Based on the foregoing, the Board should **overturn** the rejection of all pending claims and hold that all of the claims currently pending in the application under review are allowable.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 9/15/08

[illegible]

Thomas M. Coester, Reg. No. 39,637

1279 Oakmead Parkway
Sunnyvale, CA 94085-4040
(310) 207-3800

CERTIFICATE OF ELECTRONIC FILING

I hereby certify that this paper is being transmitted online via EFS Web to the Patent and Trademark Office, Commissioner for Patents, Post Office Box 1450, Alexandria, Virginia 22313-1450, on 9/15, 2008.

[illegible]

Jessica Huester 9/15, 2008

VIII. CLAIMS APPENDIX

The claims involved in this Appeal are:

1. (Original) A system comprising:

a database;

a message server having no persistent state; and

a plurality of instances of an application server implementing a Java application model coupled in a star topology with the message server at a center of the star topology, the plurality of instances sharing the database.

2. (Original) The system of Claim 1 wherein each instance comprises:

a dispatcher node; and

a plurality of server nodes.

3. (Original) The system of Claim 2 wherein each server node comprises:

a java 2 enterprise edition (J2EE) engine.

4. (Original) The system of Claim 1 further comprising:

a central lock server to provide cluster wide locks to the plurality of instances.

5. (Original) The system of Claim 1 wherein the message server comprises:

a first data structure to store a list of connected clients; and

a second data structure and a list of services provided in the system.

6. (Previously Presented) A computer readable storage media containing executable computer program instructions which when executed cause a digital processing system to perform a method comprising:

starting a central services node to provide at least one of a locking service and a messaging service, the messaging service having no persistent state;

starting a plurality of application server instances; and organizing the application server instances into a cluster having star topology with the central services node at a center of the star topology.

7. (Original) The computer readable storage media of Claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

sharing a database among the plurality of application server instances.

8. (Original) The computer readable storage media of 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method wherein starting a plurality of application server instances comprises:

starting, for each application server instance of the plurality, a dispatcher node and a plurality of server nodes.

9. (Original) The computer readable storage media of Claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

starting a message server having no persistent state.

10. (Original) The computer readable storage media of Claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

registering each application server with the messaging server.

11. (Original) The computer readable storage media of Claim 6 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

conducting inter instance communication through the messaging service.

12. (Original) The computer readable storage media of Claim 9 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

restarting the message server without state recovery responsive to a system failure.

13. (Original) The computer readable storage media of Claim 10 containing executable computer program instructions which when executed cause a digital processing system to perform the method further comprising:

notifying all registered instances from the message server when an additional instance joins the cluster.

14. (Previously Presented) A system comprising:

means for organizing a plurality of application servers instances into a cluster having a star topology with a central services node at a center of the star topology;

means for sharing a storage resource across the cluster; and

means for performing centralized inter instances communication without maintenance of persistent state information.

15. (Original) The system of Claim 14 further comprising:

means for centralized locking of a resource within the cluster.

16. (Original) The system of Claim 14 wherein the means for performing comprises:

a message server having no persistent state.

17. (Original) The system of Claim 14 wherein the means for performing comprises:

means for registering instances; and

means for recording services provided in the cluster.

18. (Previously Presented) A method comprising:

starting a central services node to provide at least one of a locking service and a messaging service, the messaging service not maintaining a persistent state;

starting a plurality of application server instances; and

organizing the application server instances into a cluster having star topology with the central services node at a center of the star topology.

19. (Original) The method of Claim 18 further comprising:

sharing a database among the plurality of application server instances.

20. (Original) The method of Claim 18 wherein starting a plurality of application server instances comprises:

starting, for each instance of the plurality, a dispatcher node and a plurality of server nodes.

21. (Original) The method of Claim 18 wherein starting a central service node comprises:

starting a message server having no persistent state.

22. (Original) The method of Claim 18 wherein organizing comprises:

registering each application server with the messaging server.

23. (Original) The method of Claim 18 further comprising:

conducting inter instance communication through the messaging service.

24. (Original) The method of Claim 21 further comprising:

restarting the message server without state recovery responsive to a system failure.

25. (Original) The method of Claim 22 wherein organizing further comprises:

notifying all registered instances from the message server when an additional instance joins the cluster.

IX. EVIDENCE APPENDIX

The evidence cited in this appeal is:

1. "Program state." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 1 July 2008. <http://en.wikipedia.org/wiki/Program_state>

2. "Message." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 28 July 2008. <http://en.wikipedia.org/wiki/Message#In_computer_science>

3. "Cache." Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 23 August 2008. <http://en.wikipedia.org/wiki/Cache>

X. RELATED PROCEEDINGS APPENDIX

There are no other appeals or interferences that will directly affect, be directly affected by, or have a bearing on the Board's decision in this appeal.

Program state

From Wikipedia, the free encyclopedia

One of the key concepts in computer programming is the idea of **state**, essentially a snapshot of the measure of various conditions in the system. Most programming languages require a considerable amount of state information in order to operate properly - information which is generally hidden from the programmer. For a real-world example, consider a three-way light switch. An ordinary switch turns on a light when it's in the "up" position, but in a three-way switch the "up" position could be on or off, depending on the state or "configuration" of the other switch, which is likely out of view.

In fact, the state is often hidden from the computer's hardware as well, which normally has no idea that this piece of information determines program state, while that piece is a temporary variable and will soon be discarded. This is a serious problem, as the state information needs to be shared across multiple processors in parallel processing machines. Without knowing which state is important and which isn't, most languages force the programmer to add a considerable amount of extra code to indicate which data and parts of the code are important in this respect.

In computer science, imperative programming is opposed to declarative programming. Imperative programming is a programming paradigm that describes computation in terms of a program state and statements that change the program state.

References

See also

- Dataflow programming
- Imperative programming
- State (computer science)

Retrieved from "http://en.wikipedia.org/wiki/Program_state"
Categories: Operating system technology
Hidden categories: Articles lacking sources from March 2007 | All articles lacking sources

Message

From Wikipedia, the free encyclopedia

A **message** in its most general meaning is an object of communication. It is something which provides information; it can also be this information itself. Therefore, its meaning is dependent upon the context in which it is used; the term may apply to both the information and its form. A **communiqué** is a brief report or statement released by a public agency.

Contents

In communications science

More precisely, in communications science, a **message** is information which is sent from a source to a receiver. Some common definitions include:

- Any thought or idea expressed in a language, prepared in a form suitable for transmission by any means of communication.
- An arbitrary amount of information whose beginning and end are defined or implied.

In communication between humans, messages can be verbal or nonverbal:

- A verbal message is an exchange of information using words. Examples include face-to-face communication, telephone calls, voicemails, etc.
- A nonverbal message is communicated through actions or behaviors rather than words. Examples include the use of body language and the actions made by an individual.

In computer science

There are two main senses of the word "message" in computer science: messages passed within software, which may or may not be human-readable, and human-readable messages delivered via computer software for person-to-person communication.

- Message passing is a form of communication used in concurrent and parallel computing, object-oriented programming, and interprocess communication, where communication is made by sending messages to recipients. In a related use of this sense of a message, in object-oriented programming languages such as Smalltalk or Java, a *message* is sent to an object, specifying a *request* for action.

- Instant messaging and e-mail are examples of computer software designed for delivering human-readable messages in formatted or unformatted text, from one person to another.

History of messaging

- Airmail
- Electronic mail
- Express mail
- Homing pigeons
- Instant messaging
- Radio
- Semaphore
- Smoke signals
- Steamshipping
- Telegraphy
- Telephony
- Television
- Text messaging
- Wind-powered shipping

External links

- A brief history of messaging through the ages (http://www.scribd.com/doc/17561/Messages-through-Ages)

Retrieved from "http://en.wikipedia.org/wiki/Message"
Categories: Communication | Units of information (cognitive processes)
Hidden categories: Articles lacking sources from December 2007 | All articles lacking sources

*Help us provide free content to the world by **donating today!***

Cache

From Wikipedia, the free encyclopedia

In computer science, a **cache** (pronounced /kæʃ/, like "cash" [1]) is a collection of data duplicating original values stored elsewhere or computed earlier, where the original data is expensive to fetch (owing to longer access time) or to compute, compared to the cost of reading the cache. In other words, a cache is a temporary storage area where frequently accessed data can be stored for rapid access. Once the data is stored in the cache, future use can be made by accessing the cached copy rather than re-fetching or recomputing the original data, so that the average access time is shorter. A cache, therefore, helps expedite data access that the CPU would otherwise need to fetch from main memory.

A cache has proven to be extremely effective in many areas of computing because access patterns in typical computer applications have locality of reference. There are several kinds of locality, but this article primarily deals with data that are accessed close together in time (temporal locality). The data might or might not be located physically close to each other (spatial locality).

Contents

History

Use of the word *cache* in the computer context originated in 1967 during preparation of an article for publication in the IBM Systems Journal. The paper concerned an exciting memory improvement in Model 85, a latecomer in the IBM System/360 product line. The Journal editor, Lyle R. Johnson, pleaded for a more descriptive term than *high-speed buffer*. When none was forthcoming, he suggested *cache*, from the French *cacher*, meaning "to hide". The paper was published in early 1968, the authors were honored by IBM, their work was widely welcomed and subsequently improved upon, and *cache* soon became standard usage in computer literature.[2]

Operation

A cache is a block of memory for temporary storage of data likely to be used again. The CPU and hard drive frequently use a cache, as do web browsers and web servers.

A cache is made up of a pool of entries. Each entry has a datum (a nugget of data) which is a copy of the datum in some backing store. Each entry also has a tag, which specifies the identity of the datum in the backing store of which the entry is a copy.

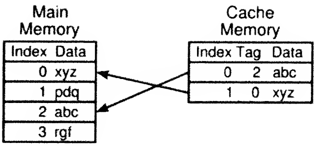

Diagram of a CPU memory cache

When the cache client (a CPU, web browser, operating system) wishes to access a datum presumably in the backing store, it first checks the cache. If an entry can be found with a tag matching that of the desired datum, the datum in the entry is used instead. This situation is known as a **cache hit**. So, for example, a web browser program might check its local cache on disk to see if it has a local copy of the contents of a web page at a particular URL. In this example, the URL is the tag, and the contents of the web page is the datum. The percentage of accesses that result in cache hits is known as the **hit rate** or **hit ratio** of the cache.

The alternative situation, when the cache is consulted and found not to contain a datum with the desired tag, is known as a **cache miss**. The previously uncached datum fetched from the backing store during miss handling is usually copied into the cache, ready for the next access.

During a cache miss, the CPU usually ejects some other entry in order to make room for the previously uncached datum. The heuristic used to select the entry to eject is known as the replacement policy. One popular replacement policy, least recently used (LRU), replaces the least recently used entry (see cache algorithms). More efficient caches compute use frequency against the size of the stored contents, as well as the latencies and throughputs for both the cache and the backing store. While this works well for larger amounts of data, long latencies, and slow throughputs, such as experienced with a hard drive and the Internet, it's not efficient to use this for cached main memory (RAM).

When a datum is written to the cache, it must at some point be written to the backing store as well. The timing of this write is controlled by what is known as the **write policy**.

In a **write-through** cache, every write to the cache causes a synchronous write to the backing store.

Alternatively, in a **write-back** (or **write-behind**) cache, writes are not immediately mirrored to the store. Instead, the cache tracks which of its locations have been written over (these locations are marked **dirty**). The data in these locations is written back to the backing store when those data are evicted from the cache, an effect referred to as a **lazy write**. For this reason, a miss in a write-back cache (which requires a block to be replaced by another) will often require two memory accesses to service: one to retrieve the needed datum, and one to write replaced data from the cache to the store.

Data write-back may be triggered by other policies as well. The client may make many changes to a datum in the cache, and then explicitly notify the cache to write back the datum.

No-write allocation is a cache policy where only processor reads are cached, thus avoiding the need for write-back or write-through when the old value of the datum was absent from the cache prior to the write.

The data in the backing store may be changed by entities other than the cache, in which case the copy in the cache may become out-of-date or **stale**. Alternatively, when the client updates the data in the cache, copies of that data in other caches will become stale. Communication protocols between the cache managers which keep the data consistent are known as coherency protocols.

Applications

CPU caches

Small memories on or close to the CPU chip can be made faster than the much larger main memory. Most CPUs since the 1980s have used one or more caches, and modern general-purpose CPUs inside personal computers may have as many as half a dozen, each specialized to a different part of the task of executing programs.

Disk cache

While CPU caches are generally managed entirely by hardware, other caches are managed by a variety of software. The page cache in main memory, which is an example of disk cache, is usually managed by the operating system kernel.

While the hard drive's hardware disk buffer is sometimes misleadingly referred to as "disk cache", its main functions are write sequencing and read prefetching. Repeated cache hits are relatively rare, due to the small size of the buffer in comparison to HDD's capacity.

In turn, fast local hard disk can be used to cache information held on even slower data storage devices, such as remote servers (web cache) or local tape drives or optical jukeboxes. Such a scheme is the main concept of hierarchical storage management.

Other caches

The BIND DNS daemon caches a mapping of domain names to IP addresses, as does a resolver library.

Write-through operation is common when operating over unreliable networks (like an Ethernet LAN), because of the enormous complexity of the coherency protocol required between multiple write-back caches when communication is unreliable. For instance, web page caches and client-side network file system caches (like those in NFS or SMB) are typically read-only or write-through specifically to keep the network protocol simple and reliable.

A cache of recently visited web pages can be managed by your web browser. Some browsers are configured to use an external proxy web cache, a server program through which all web requests are routed so that it can cache frequently accessed pages for everyone in an organization. Many internet service providers use proxy caches to save bandwidth on frequently-accessed web pages.

Search engines also frequently make web pages they have indexed available from their cache. For example, Google provides a "Cached" link next to each search result. This is useful when web pages are temporarily inaccessible from a web server.

Another type of caching is storing computed results that will likely be needed again, or memoization. An example of this type of caching is ccache, a program that caches the output of the compilation to speed up the second-time compilation.

Database caching can substantially improve the throughput of database applications, for example in the processing of indexes, data dictionaries, and frequently used subsets of data. TimesTen provides a mid-tier caching facility that can be integrated into Oracle databases.

The difference between buffer and cache

The terms are not mutually exclusive and the functions are frequently combined; however, there is a difference in intent. A buffer is a temporary memory location, that is traditionally used because CPU instructions cannot directly address data stored in peripheral devices. Thus, addressable memory is used as intermediate stage. Additionally such a buffer may be feasible when a large block of data is assembled or disassembled (as required by a storage device), or when data may be delivered in a different order than that in which it is produced. Also a whole buffer of data is usually transferred sequentially (for example to hard disk), so buffering itself sometimes increases transfer performance. These benefits are present even if the buffered data are written to the buffer once and read from the buffer once.

A cache also increases transfer performance. A part of the increase similarly comes from the possibility that multiple small transfers will combine into one large block. But the main performance gain occurs because there is a good chance that the same datum will be read from cache multiple times, or that written data will soon be read. Cache's sole purpose is to reduce accesses to the underlying slower storage. Cache is also usually an abstraction layer that is designed to be invisible from the perspective of neighboring layers.

References

1. ^ or, formerly, /k ː ʃ/ — Oxford English Dictionary *cache* (http://dictionary.oed.com/cgi/entry/50030859?query_type=word&queryword=cache&first=1&max_to_show=10&sort_type=alpha&result_place=1&search_id=ckqc-Vj9VUU-13102&hilite=50030859) (restricted); also Dictionary.com (http://dictionary.reference.com/search?q=cache) (unrestricted). Although the pronunciation /ˈkæʃeɪ/ is sometimes heard in English, it properly only represents the French adjective 'caché(e)', meaning 'hidden'. /keɪʃ/ is a mispronunciation.
2. ^ G. C. Stierhoff and A. G. Davis. A History of the IBM Systems Journal. *IEEE Annals of the History of Computing*, Vol. 20, No. 1 (Jan. 1998), pages 29-35. [1] (http://dx.doi.org/10.1109/85.646206)

See also

- Disk buffer (Hardware-based cache)
- Cache algorithms
- Cache coloring
- Caching failure
- CPU cache
- Web cache
- Data grid

Retrieved from "http://en.wikipedia.org/wiki/Cache"
Categories: Cache | Central processing unit | Computer memory | Internet

Hidden categories: All articles with unsourced statements | Articles with unsourced statements since May 2007